MINISTRY OF SCIENCE AND EDUCATION
NATIONAL TECHNICAL UNIVERSITY
“KHARKIV POLYTECHNIC INSTITUTE”
DEPARTMENT OF SOFTWARE ENGINEERING AND MANAGEMENT
INFORMATION TECHNOLOGIES

METHODICAL RECOMMENDATION TO

“BASICS OF SOFTWARE ENGINEERING
LABORATORY PRACTICE
PART 2”

For students of specialties
121 “Software Engineering”,
122 “Computer Science and Intellectual Systems”

Kharkiv
NTU “KhPI”
2019

Methodical recommendation to “Basics of software engineering. Laboratory practice. Part 1” for students of Technical science / authors Melnyk K.V., Borisova N.V., Lutenko I.V., Ershova S.I., Smolin P.A., Grinchenko M.A. – Kharkiv : NTU “KhPI”. – 20 p.

Authors Melnyk K.V.,
Borisova N.V.,
Lutenko I.V.,
Ershova S.I.,
Smolin P.A.,
Grinchenko M.A.

Reviewer Shmatko A.V.

Department of Software Engineering and Management Information Technologies

CONTENT

INTRODUCTION

Software engineering is an engineering discipline that is concerned with all aspects of software production [1].

Software engineering can be divided into sub-disciplines [2]. Some of them are:

- Software engineering management: The application of management activities – planning, coordinating, measuring, monitoring, controlling, and reporting – to ensure that the development and maintenance of software is systematic, disciplined, and quantified.
- Requirements engineering: The elicitation, analysis, specification, and validation of requirements for software.
- Software design: The process of defining the architecture, components, interfaces, and other characteristics of a system or component.
- Software construction: The detailed creation of working, meaningful software through a combination of programming, verification, unit testing, integration testing, and debugging.
- Software testing: An empirical, technical investigation conducted to provide stakeholders with information about the quality of the product or service under test.
- Software maintenance: The totality of activities required to provide cost-effective support to software.
- Software quality.

The outcome of software engineering is an efficient and reliable software product. Software product is a computer programs with all associated documentation and configuration data that is required to make these programs operate correctly [1]. Essential attributes of good software product: maintainability, dependability and security, efficiency, acceptability.

Authors: Melnyk K.V., Borisova N.V., Lutenko I.V., Ershova S.I., Smolin P.A., Grinchenko M.A.
Department of SEMIT, NTU «KhPI»
Kharkiv, 2019

A software process is a sequence of activities that leads to the production of a software product. There are four fundamental activities that are common to all software processes. These activities are:

1. Software specification, where customers and engineers define the software that is to be produced and the constraints on its operation.

2. Software development, where the software is designed and programmed.

3. Software validation, where the software is checked to ensure that it is what the customer requires.

4. Software evolution, where the software is modified to reflect changing customer and market requirements.

The methodical recommendation is about modelling (or designing) and creating a software that will help to calculate some system of expressions with unknown variables. The values of variables can be obtained from different sources: from file or from keyboard.

Authors: Melnyk K.V., Borisova N.V., Lutenko I.V., Ershova S.I., Smolin P.A., Grinchenko M.A.
Department of SEMIT, NTU «KhPI»
Kharkiv, 2019

LAB № 3

PLANNING OF SYSTEM ACTIVITIES

Goal: To obtain basic experience and knowledge to work with project tracking system

Tasks:

1. Register on the free xp-dev.com account.
2. Create new project.
3. Add new user to project.
4. Create Iterations and user stories according to previous labs. Add task to implementation of user story. Assign tasks to users of you team.
5. Work with svn repository.
6. Prepare the report of the work.

Progress of the lab.

1. Register on the free xp-dev.com account

This task should execute only one person from team.

In order to register you should have an email account (it is also possible to provide a free email account). In order to register go to https://xp-dev.com/register and create new account (figure 3.1). The account provides you with 2Gb of storage for the repository and 30 days of trial use.

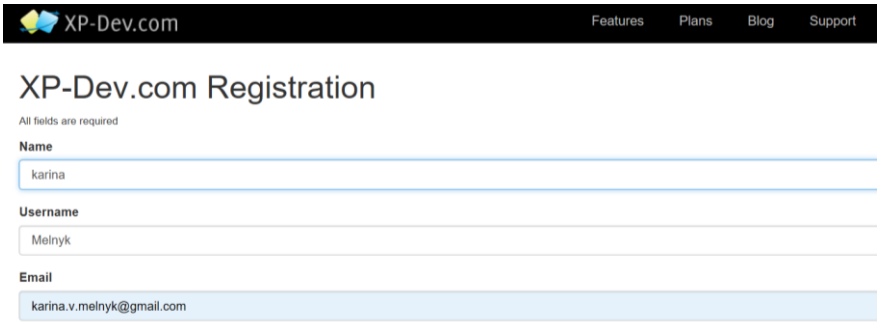

Figure 3.1 – Registering on xp-dev.com

Authors: Melnyk K.V., Borisova N.V., Lutenko I.V., Ershova S.I., Smolin P.A., Grinchenko M.A.
Department of SEMIT, NTU «KhPI»
Kharkiv, 2019

2. Create new project

This task should execute only one person from team.

Choose “Create new project" and give a name, abbreviation and description for the project Issue, tracking Type should be XPDev (figure 3.2). It is necessary to create 1 xpdev account for 1 team. Create new project Inside XpDev.

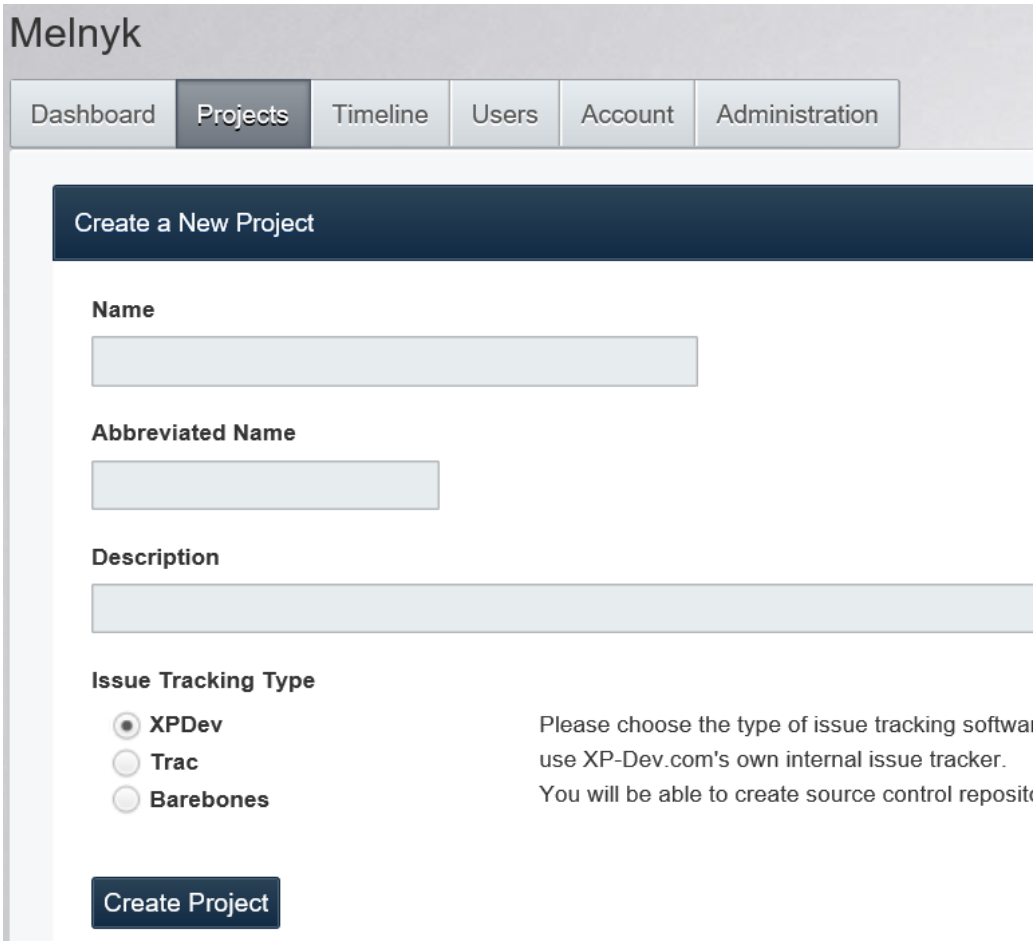

Figure 3.2 – Creating of new project

3. Add new user to project.

Choose the username and password to send invitation (figure 3.3). New user can log in with the credentials sent to his/her email account. Choose the role of user (examine all roles first). The administrator or writer are acceptable.

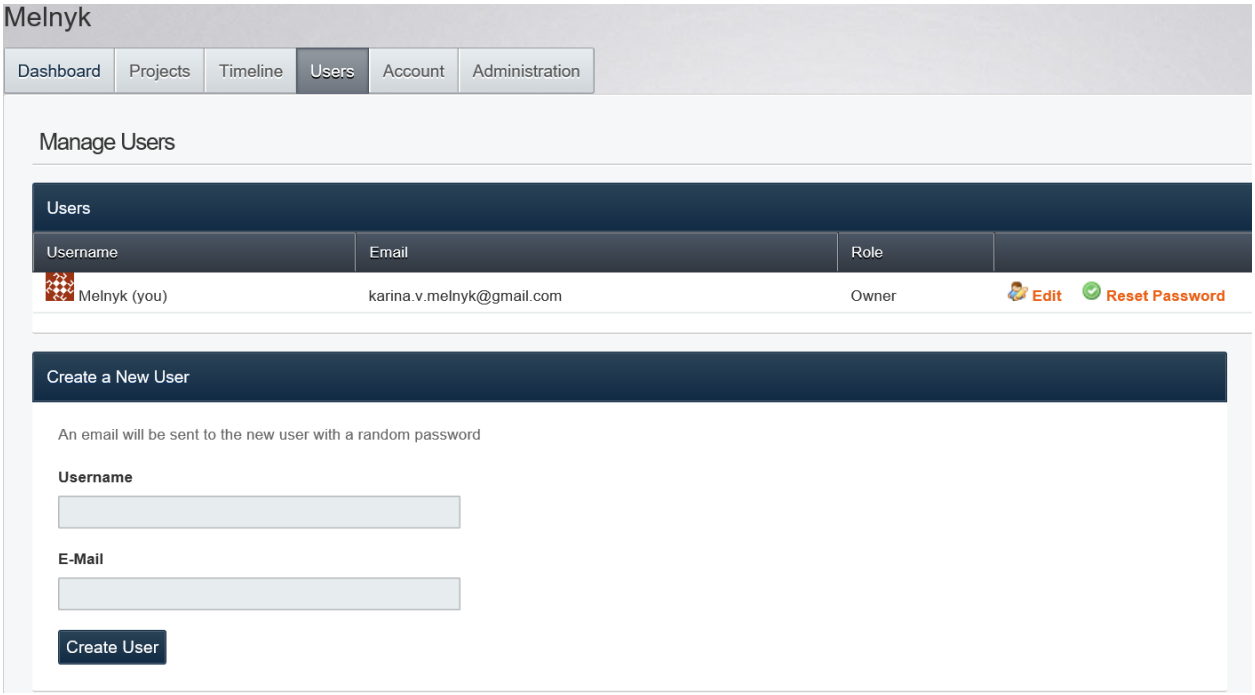

Figure 3.3 – Adding second person to the team

4. Create Iterations and user stories according to previous labs. Add task to implementation of user story. Assign tasks to users of you team.

The modern software development approaches plan the project activities by stages (iteration). Every iteration should implement several software or system functionalities (user stories). Therefore, every laboratory start with new iteration that can have several user stories.

Go to Project tracking and choose "Create new iteration". Name: any of one that describe setup process. Choose dates for iteration (figure 3.4). Create a story for each iteration (figure 3.5). Create a set of tasks for each story (figure 3.6).

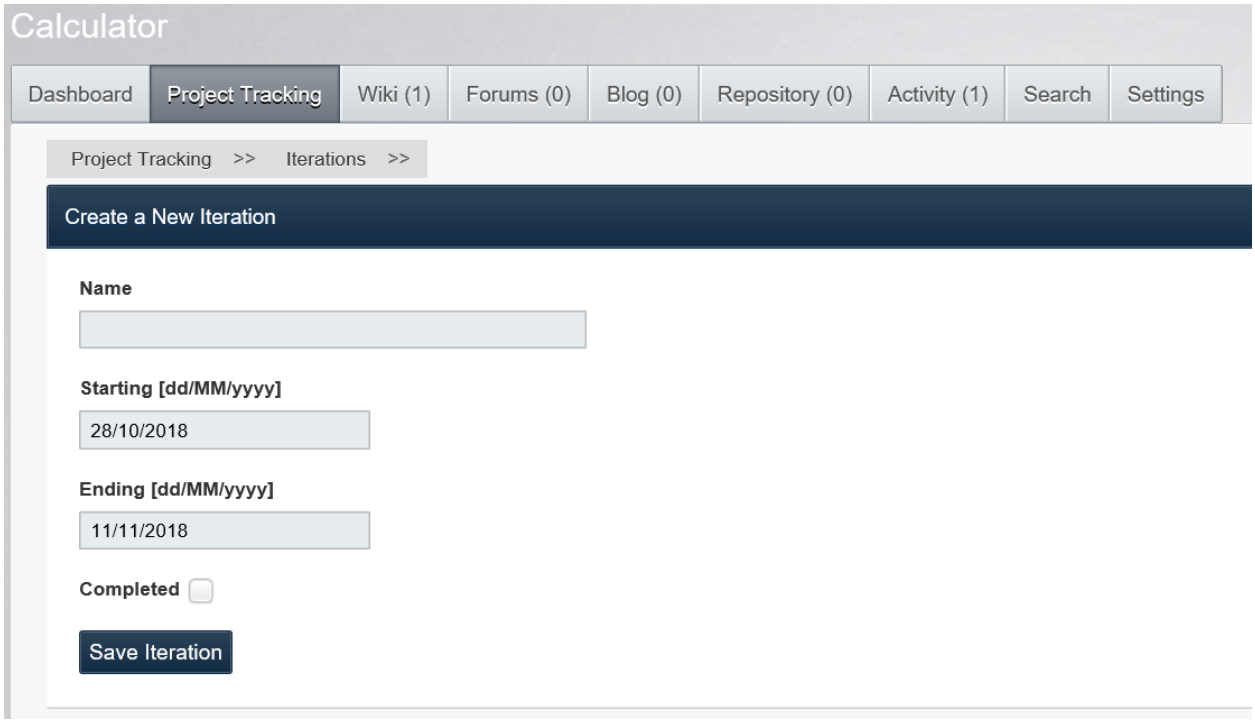

Figure 3.4 – Creating a new iteration

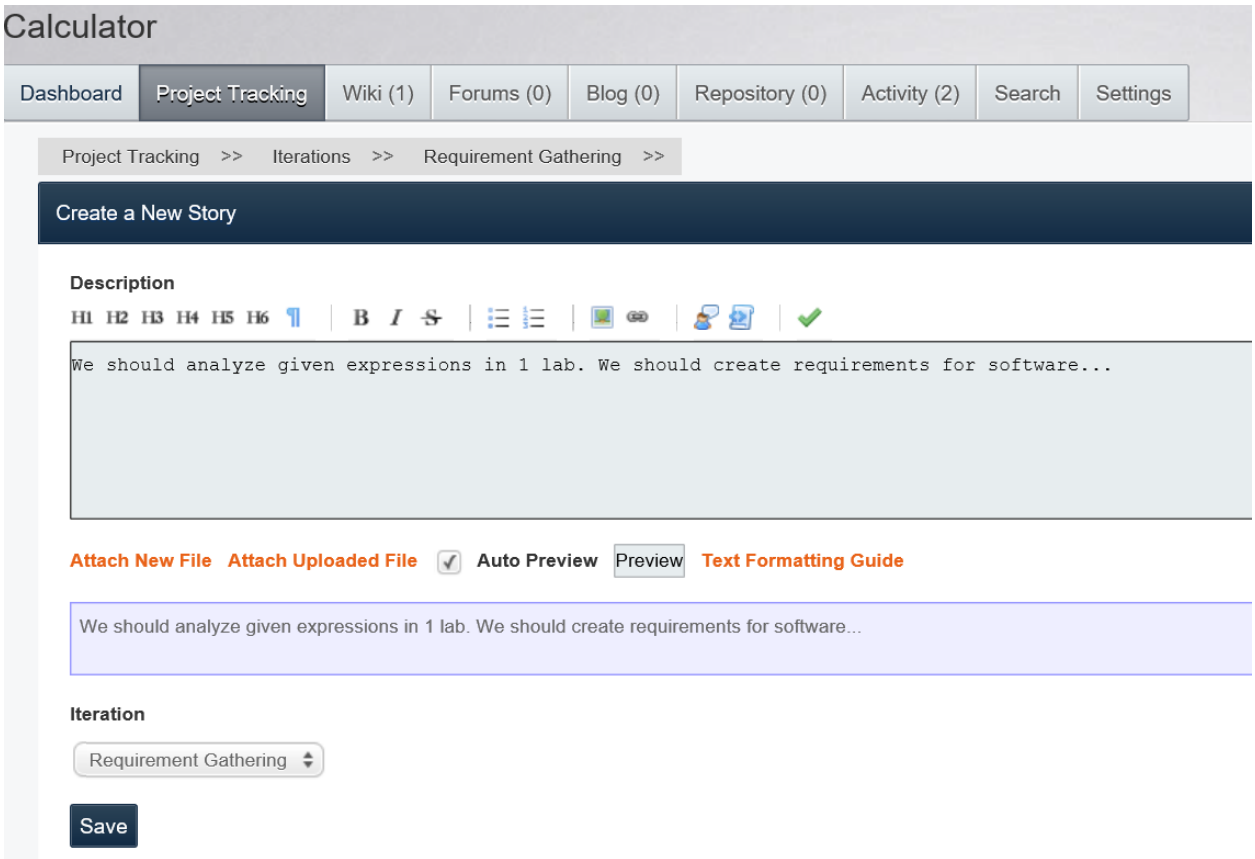

Figure 3.5 – Creating a new story

Authors: Melnyk K.V., Borisova N.V., Lutenko I.V., Ershova S.I., Smolin P.A., Grinchenko M.A.
Department of SEMIT, NTU «KhPI»
Kharkiv, 2019

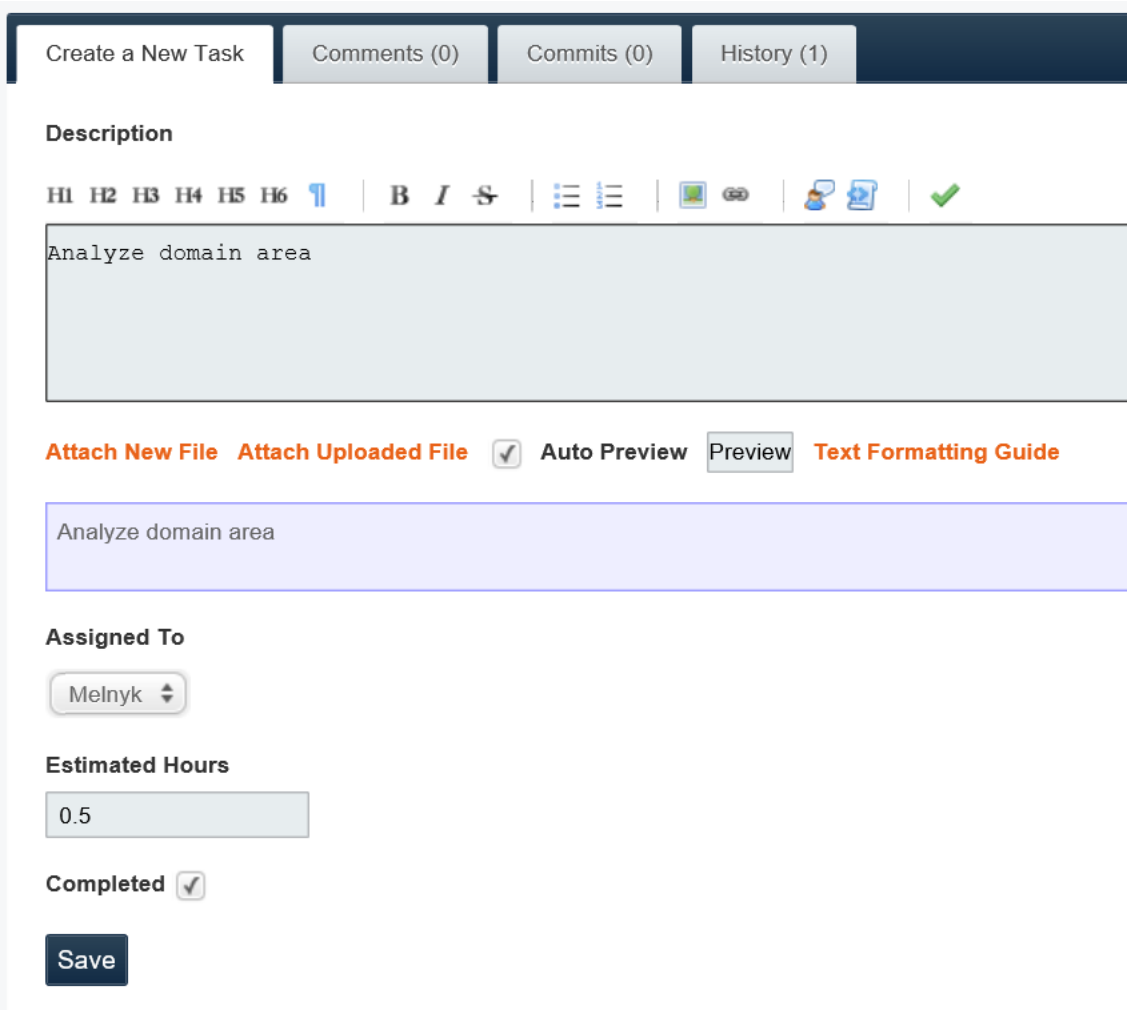

Figure 3.6 – Creating a new task

Choose any software development lifecycle model for your software. For example: you have chosen waterfall model, than iterations in your projects are stages from waterfall model (table 3.1).

Table 3.1 – Example of iterations

Iterations	User story	Tasks
1. Requirement Gathering	We should analyze given expressions in 1 lab. We should create requirements for software...	1. Analyze domain area
		2. Write software requirement specification
		3. Draw Use-case diagram
2. System Analysis	…	…

The result table with all iterations should be in report, example is shown on figure 3.7. There are many tables with set of tasks for each iteration in report (figure 3.8).

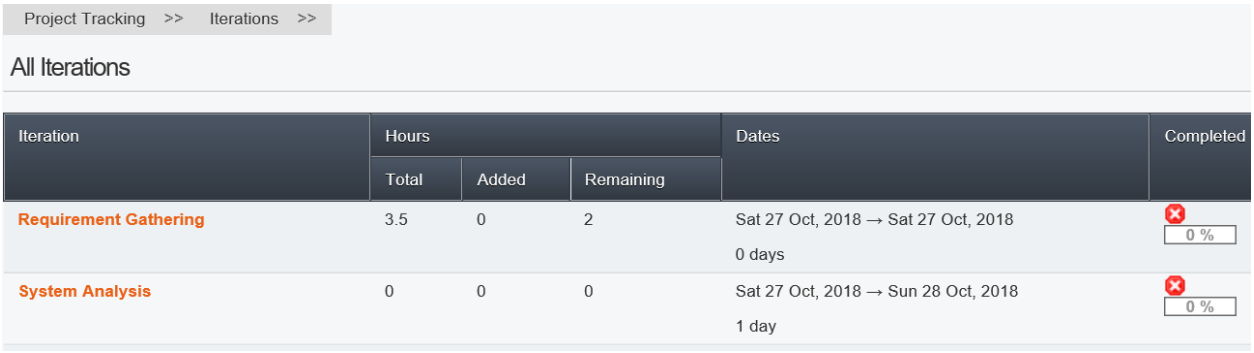

Figure 3.7 – Table with all iterations

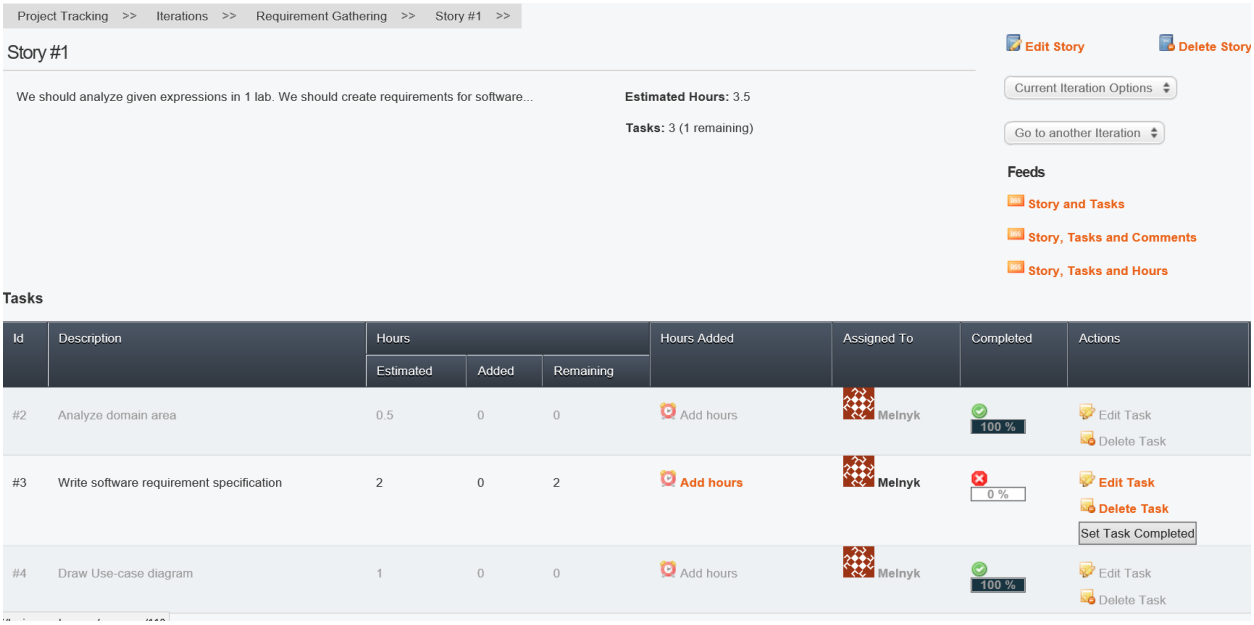

Figure 3.8 – Result table for the first iteration

5. Work with svn repository.

Create a Repository at xp-dev.com : Choose Tab “Repository”, then create a new repository. Set the name of repository. Set Create Initial Directories switch (it

will creates additional directories to handle multiple version of project to handle customer requirements changes). Result of these actions is shown on figure 3.9.

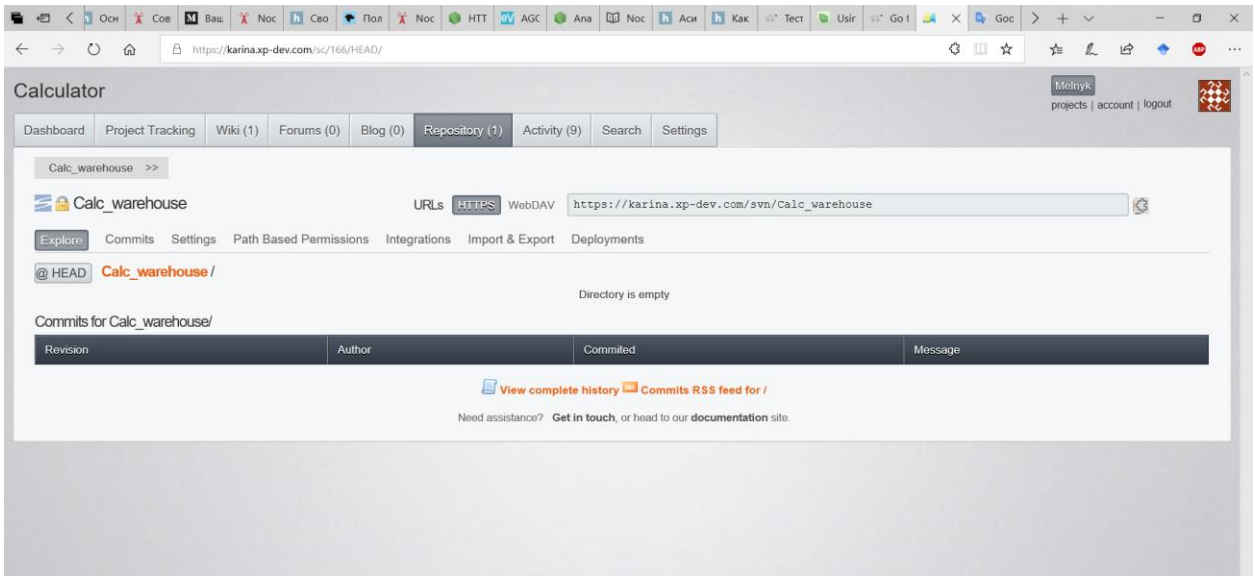

Figure 3.9 – A new repository on xp-dev.com

Open work directory on PC with *.cpp file from previous lab. Now right click on the file and from the context menu choose TortoiseSVN → Repo-browser (figure 3.10). After that type in link with your URL of repository from xp-dev.com (figure 3.11).

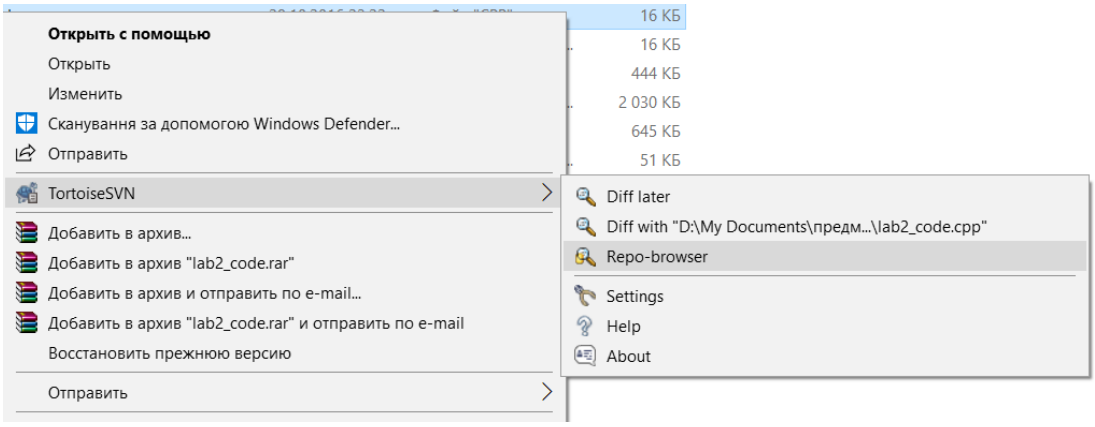

Figure 3.10 – A popup menu of TortoiseSVN

Authors: Melnyk K.V., Borisova N.V., Lutenko I.V., Ershova S.I., Smolin P.A., Grinchenko M.A.
Department of SEMIT, NTU «KhPI»
Kharkiv, 2019

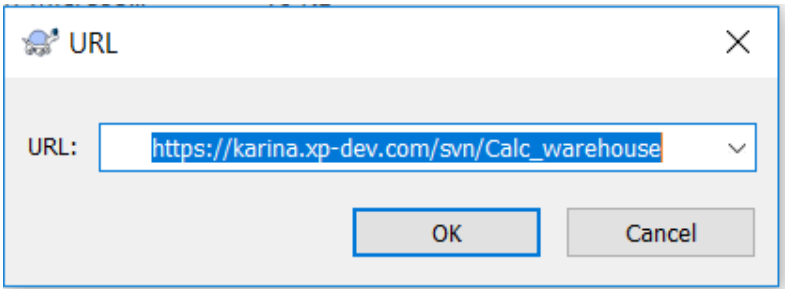

Figure 3.11 – URL of repository from xp-dev.com

You must add file to repository: right click on Tortoise window, then choose "Add file" (figure 3.12). Result of these actions is shown on figure 3.13.

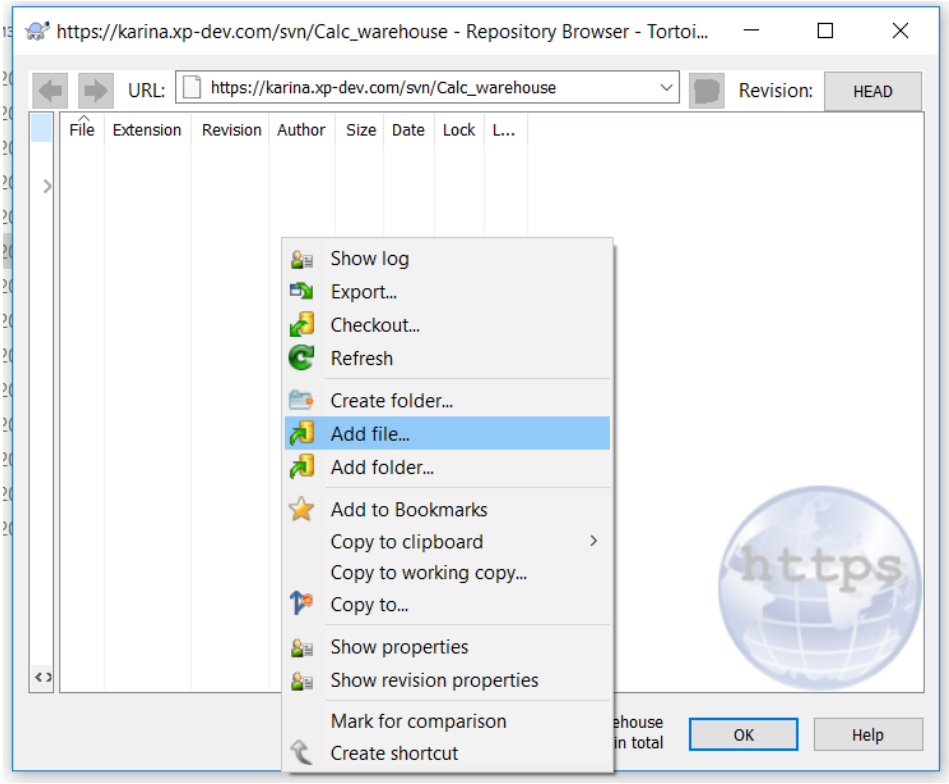

Figure 3.12 – Adding a new file to xp-dev.com repository

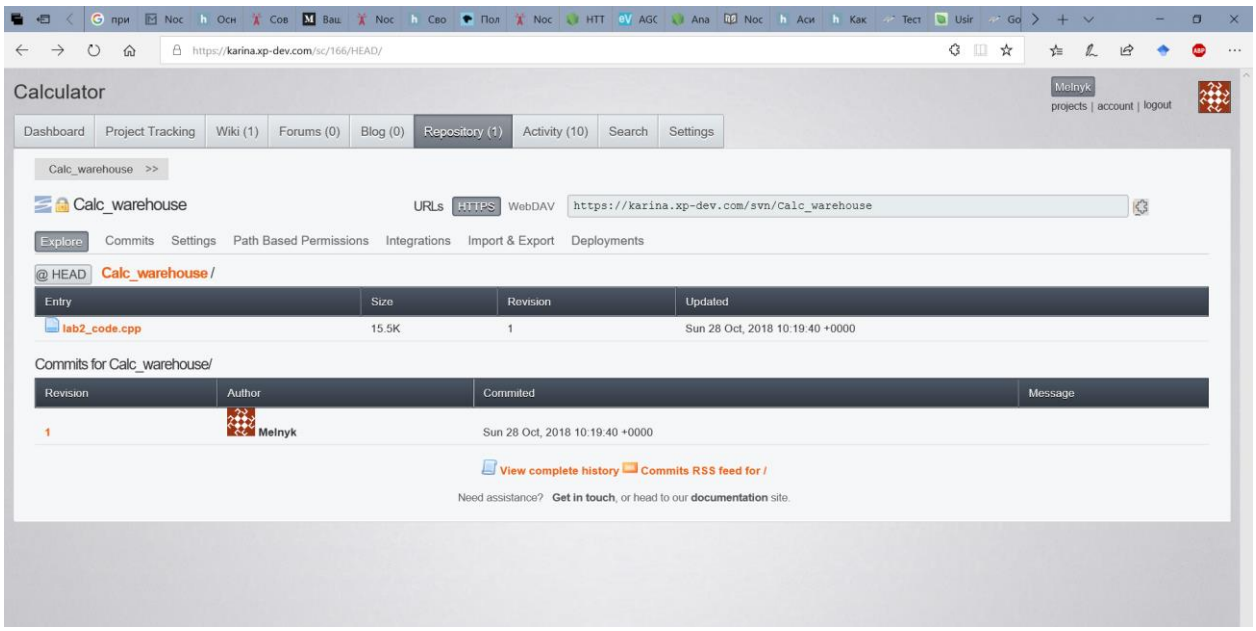

Figure 3.13 – File in repository

If the team member want to make some changes in some files, he/she should make “checkout” for necessary files (figure 3.14) and set the appropriate working directory (figure 3.15).

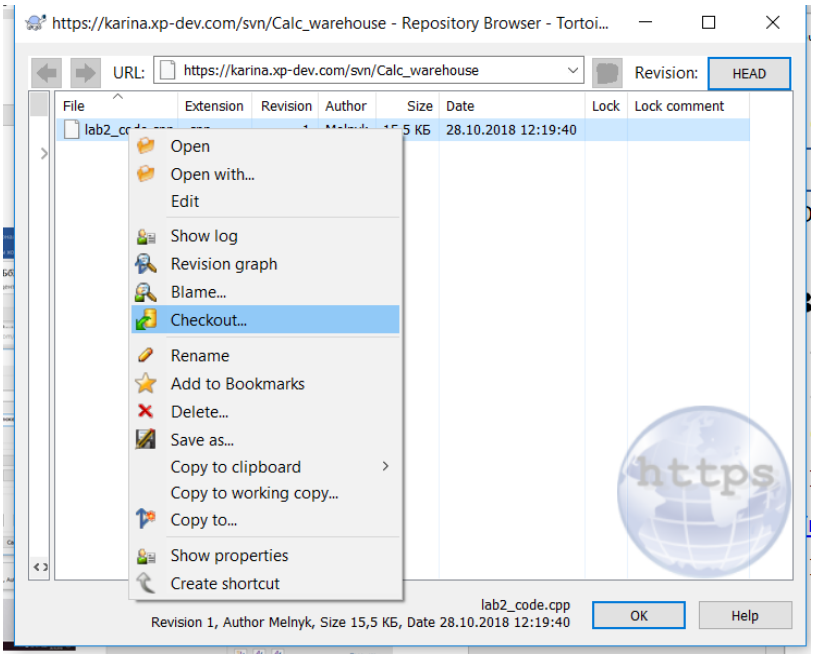

Figure 3.14 – Checkouting of file

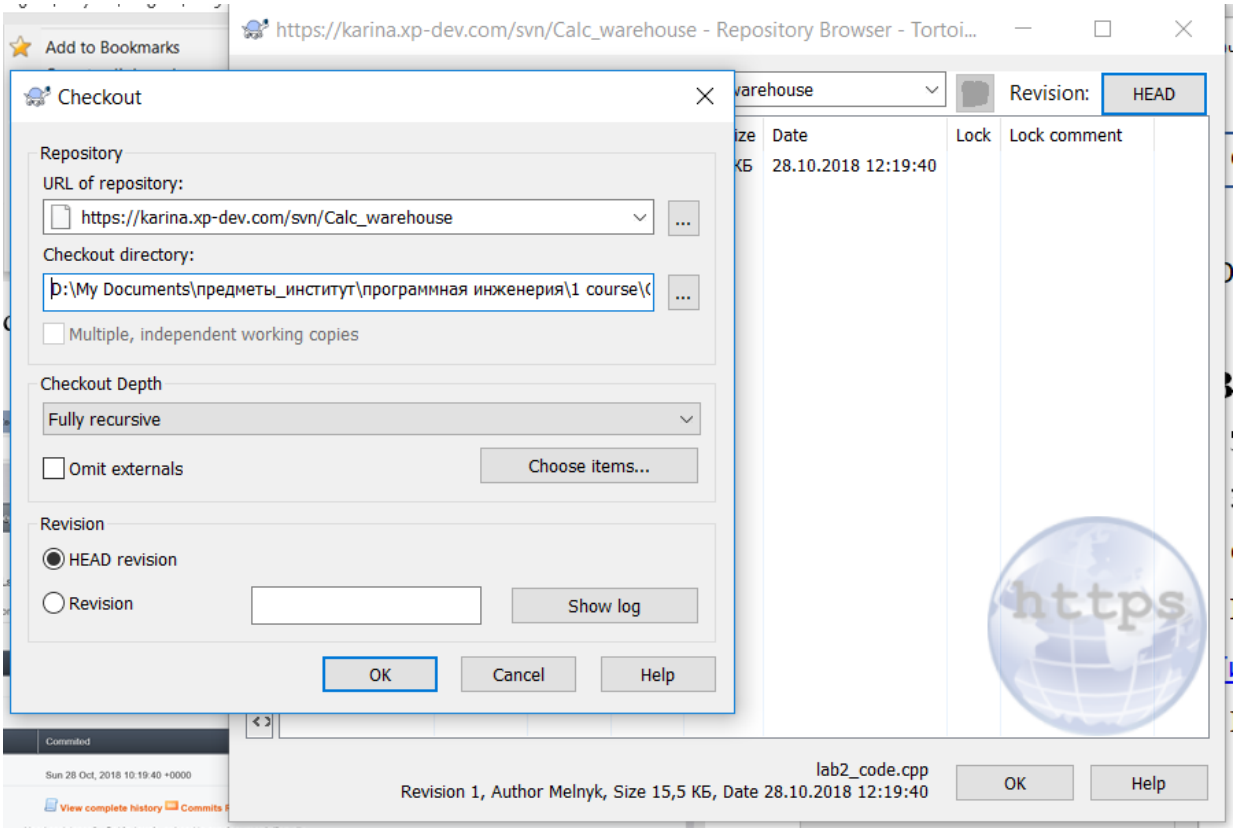

Figure 3.15 – Working directory for file

Result of these actions is shown on figure 3.16.

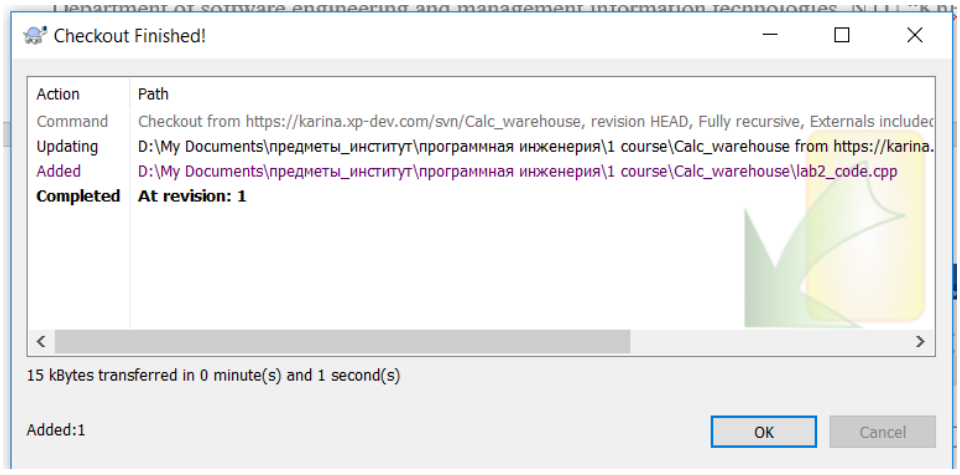

Figure 3.16 – Result of checkout process

When the file has been modified, the file should be committed to xp-dev.com repository. Press right click on file, choose “SVN Commit”, check file, press Ok (figure 3.17).

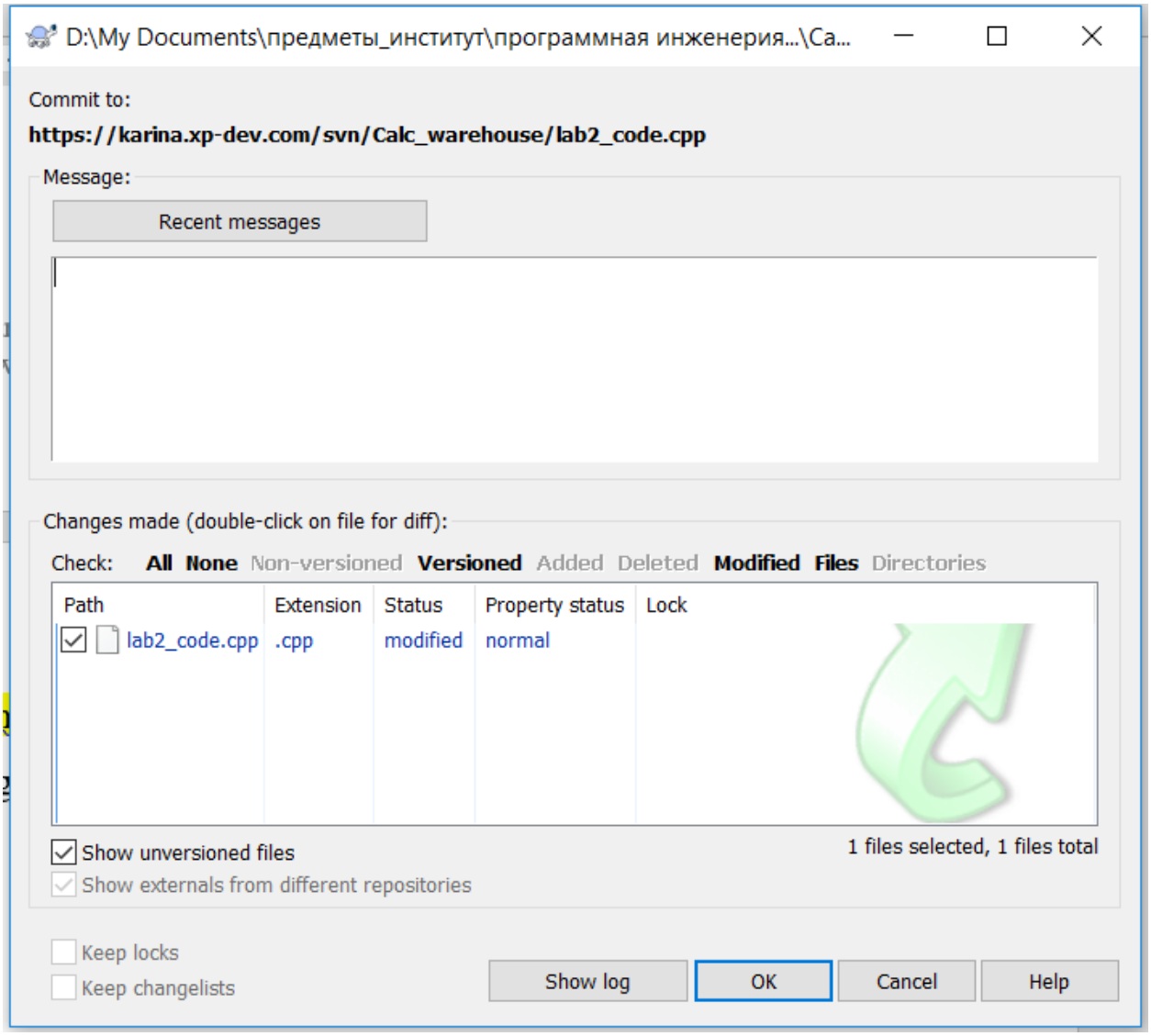

Figure 3.17 – Process of committed file

6. Prepare the report of the work

The report shall contain the results of implementation of all the tasks of the lab in accordance with the individual task.

LAB № 4

CREATING OF SOFTWARE REQUIREMENT SPECIFICATION

Goal: Learning basic principles of creating Software Requirement Specification

Tasks:

1. Choose some pattern for creating SRS.
2. Fill in the SRS according previous labs.
3. Make all necessary actions on xp-dev.com. Show the iteration where you made updating information.
4. Prepare the report of the work.

Progress of the lab.

1. Choose some pattern for creating SRS.

A Software Requirements Specification (SRS) is a document that describes software or application. This document is also known by the names SRS report, software document. A software document is primarily prepared for a software or any kind of application. There are a set of guidelines to be followed while preparing the software requirement specification document. This includes the purpose, scope, functional and nonfunctional requirements, software and hardware requirements of the project. In addition to this, it also contains the information about environmental conditions required, safety and security requirements, software quality attributes of the project etc.

There are many patterns for Software Requirement Specifications. You may use the following pattern of SRS:

1. Purpose
    a. Definitions
    b. System overview

Authors: Melnyk K.V., Borisova N.V., Lutenko I.V., Ershova S.I., Smolin P.A., Grinchenko M.A.
Department of SEMIT, NTU «KhPI»
Kharkiv, 2019

   c. References
2. Overall description
   a. Product perspective
      i. System Interfaces
      ii. User interfaces
      iii. Hardware interfaces
      iv. Software interfaces
      v. Communication Interfaces
      vi. Memory Constraints
   b. Design constraints
      i. Operations
      ii. Site Adaptation Requirements
   c. Product functions
   d. User characteristics
   e. Constraints, assumptions and dependencies
3. Specific requirements
   a. External interface requirements
   b. Functional requirements
   c. Performance requirements
   d. Logical database requirement
   e. Software System attributes
      i. Reliability
      ii. Availability
      iii. Security
      iv. Maintainability
   f. Environment characteristics
      i. Hardware
      ii. peripherals

Another example of SRS pattern is shown on figure 4.1.

1. Introduction

1.1 Purpose

1.2 Document Conventions

1.3 Intended Audience and Reading Suggestions

1.4 Project Scope

1.5 References

2. Overall Description

2.1 Product Perspective

2.2 Product Features

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 Assumptions and Dependencies

3. System Features

3.1 Functional Requirements

4. External Interface Requirements

4.1 User Interfaces

4.2 Hardware Interfaces

4.3 Software Interfaces

4.4 Communications Interfaces

5. Nonfunctional Requirements

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

Figure 4.1 – SRS pattern

2. Fill in the SRS according previous labs.

Your task is creating software for calculating some system of expressions. Therefore, you should fill in chosen pattern according to previous labs.

3. Make all necessary actions on xp-dev.com. Show the iteration where you made updating information

In this lab we have created SRS, so we can set task “Write Software Requirements Specification” as completed. This action is shown on figure 4.2.

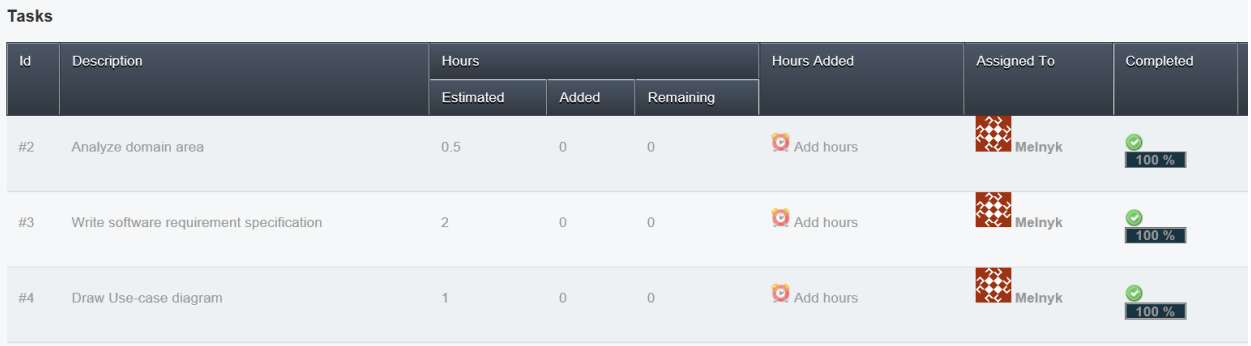

Figure 4.2 – Completed task

4. Prepare the report of the work

Make a report with all actions according to the tasks from this lab.

REFERENCE LIST

1. Sommerville I. Software engineering / I. Sommerville // Boston: Pearson, 2011. ISBN 0-13-705346-0.
2. IEEE Computer Society. Software Engineering Body of Knowledge (SWEBOK Version 3) [Electronic resource] / Mode of access: www.swebok.org – 12.11.2018.
3. Richard L. Halterman. Fundamentals of Programming C++ / Richard L. Halterman // Publisher: Southern Adventist University, 2018. – 742 p.
4. Jon Kalb. Title C++ Today: The Beast Is Back / Jon Kalb, Gasper Azman // ISBN-13: 978-1491931660. – Publisher: O'Reilly Media, 2015. – 74 p.

Authors: Melnyk K.V., Borisova N.V., Lutenko I.V., Ershova S.I., Smolin P.A., Grinchenko M.A.
Department of SEMIT, NTU «KhPI»
Kharkiv, 2019